ViSta

The Visual Statistics System

How to Use ViSta

Forrest W. Young

The Visual Statistics Project

www.visualstats.org

FORREST W. YOUNG
PSYCHOMETRIC LABORATORY
UNIVERSITY OF NORTH CAROLINA
CB 3270 DAVIE HALL
CHAPEL HILL, NC 27599-3270 USA

PEDRO VALERO
INSTITUTE OF TRAFFIC AND ROAD SAFETY
UNIVERSITAT DE VALÈNCIA
C/ HUGO DE MONCADA 4. ENTRESUELO.
VALENCIA, 46010, ESPAÑA (SPAIN)

JANUARY, 2001 - VISTA 6.1

How To Use ViSta

When you are learning how to use ViSta, you should turn to its help system: ViSta has an extensive help system, covering all aspects of using ViSta, from specific to general. It is structured into a set of general overviews and a set of specific context-dependent topics.

The general overviews are presented in this report. You can access these same materials while using ViSta by choosing the HELP TOPICS item of the HELP menu. You will then see ViSta's Help Panel which you can use to view the materials that are printed here. The specific context-dependent details can be obtained from the help buttons and menu items found throughout ViSta.

Young, Forrest W. "How To Use ViSta", January, 2001. The Visual Statistics Project. Chapel Hill, NC, USA and Valencia, Spain.

ViSta was created, designed and primarily implemented by Forrest W. Young, with significant additional code by Pedro Valero. You can contact us at

forrest@visualstats.org pedro@visualstats.org

There are two news groups about ViSta. One is for users who wish to find out more about ViSta or who have news about ViSta that you think others would be interested in. The other newsgroup is for developers who are interested in contributing to ViSta, which is an open-software project. The addresses are:

vista@visualstats.org developers@visualstats.org

If you have a bug report, consult the buglist at visualstats.org. If your bug is not on the list, then please send it to:

bugs@visualstats.org

How to Use ViSta

Page

Table of Contents

1 **FRONTMATTER** - Information about ViSta, including who the authors and developers are, how to contact us, and what the copyrights are. There is also a poetic description of data analysis (!) and a section acknowledging those who lent a hand.

5 **WELCOME TO VISTA** - Welcomes you to ViSta with an overview of our new Help system and of our other new major features.

17 **USING THE DESKTOP** - Overview of ViSta's desktop, including its options, workmap, toolbar, datsheet, selector and listener.

31 **USING THE MENUS** - Introduces you to the layout and function of ViSta's DeskTop menu system, with special emphasis on the Options and Window menus, and on how to get help for individual menu items.

39 **EXCEL AND VISTA** - How to use ViSta to visualize and analyze your Excel data, and how to customize the kinds of visualizations and analyses Excel asks ViSta to do.

43 **ENTERING DATA** - Covers data entry topics for ViSta, including entering data into the datasheet, importing data, simulating data and writing datacode to enter data and to modify your data so that it is suitable for further analysis.

49 **MANAGING DATA** - These topics include an overview of ViSta's data system, with focus on ViVa (ViSta's Interactive Variable Application language for manipulating data), VAR (ViSta's function for creating variables) and DATASET (ViSta function for creating data objects).

first, you see your data for what they seem to be
then, you ask them for the truth -
are you what you seem to me?

you see with broad expanse
yet ask with narrowed power
you see and ask and see
and ask and see ... and ask

with brush you paint the possibilities
with pen you scribe the probabilities

for in pictures we find insight
while in numbers we find strength

Acknowledgements

Over the last 10 years, ViSta has evolved from a testbed for the design of a highly interactive, very dynamic statistical visualization system, to just such a system. As the creator, designer and primary implementor of ViSta, I am deeply indebted to many, many people who have contributed in one way or another over the years. Without them, ViSta would not exist.

My deepest thanks and appreciation go to my wife Patricia Young. During the last decade she has stood watch while I wandered, lost in the developer's cave, evolving ViSta from what I could see on the screen to what I could see in my mind's eye. Patty's patience, understanding, forebearance, support, and steady love, have been crucial to the realization of my dreams. Her artistic talent, abilities and spirit have shaped those dreams, and have had a deep impact on ViSta's design.

Mucho Gracias to Pedro Valero-Mora for his time, energy, enthusiasm and, especially, his code. Pedro wrote the code for missing data features, including visual transformation and imputation methods; as well as the visually-based Box-Cox and Folded Power transformations, and is working on log-linear modeling. I expect exciting new developments to come.

I greatly appreciate Dominic Moore's friendship, and support. His encouragement, criticisms, cheers, enthusiasm, pep-talks, humor, luniness, food, music and incredible speakers are food for my soul.

Nathan Vandergrift provided a "joie de vie" at a critical juncture late in the project and has reminded me of potentials in this project that I had long forgot. Thanks for renewing the faith.

Kuddos and congratulations to Luke Tierney for creating and providing the XLisp-Stat statistical development system, which is the foundation on which ViSta is built. And thanks to the many folks who have built the XLisp system on which XLisp-Stat was itself built. A great example of Open Software at its best.

Congratulations to Sandy Weisberg. Your timely and insightful suggestions on getting ViSta out the door have helped, though perhaps not as much as everyone hoped. There's no-one else who understands LispStat the way we do.

David Lubinsky and John B. Smith made very fundamental contributions to WorkMap architecture. Without their seminal input there'd be no ViSta. Richard Faldowski and Carla Bann contributed fundamentally to SpreadPlot architecture, and spent many intense months implementing their ideas.

"Merci Beaucoup" to Louis Le Guelte for the French translation, and "Mucho Gracias" to Maria R. Rodrigo, Gabriel Molina and Pedro Valero for the Spanish translation.

Great appreciation to Bibb Latane, of Social Science Conferences, Inc. for funding development of the Excel-ViSta connection and multipane windows, and for funding the first developer's conference. Angell Beza, Associate Director of UNC's Institute for Research in the

Social Sciences, was instrumental in obtaining the SSCI funds, which supported Fabian Comacho's implementation.

Doug Kent, funded by UNC's Office of Information Technology, implemented Version 5's MS-Windows features. Jenny Williams of UNC's Office of Information Technology, was my NT-Guru, especially on the network installation.

Mosaic Plots and Bar Charts are based on an algorithm by Ernest Kwan. Frequency Polygons and Histograms based on algorithms by Jan de Leeuw and Jason Bond. Many of the Model Objects were written by Carla Bann, Rich Faldowski, David Flora, Ernest Kwan, Lee Bee Leng, Mary McFarland and Chris Weisen.

The C-to-Lisp Parser used in ViVa was written and copyrighted by Erann Gat, who reserves all rights. Used under the terms of the GNU General Public License. The developer's menu derives from Kjetil Halvorsen, the Symbol Editor Dialog was implemented by Frederic Udina, and the BitMap Editor by Fabian Camacho. Code for earlier versions was modified for Unix by Anthony J. Rossini, Charles Kurak, Albrecht Gebhardt and Andrew V. Klein.

Quality assurance provided, unwittingly, by my undergraduate students, who suffered through "Creaping-Featuritis" and old documentation.

Thanks to all of you for helping me make ViSta what it is today. Without you, it would not exist.

Forrest

ViSta - The Visual Statistics System
Wednesday, December 13, 2000 - 8:12 AM

Welcome to ViSta - The Visual Statistics System (1 of 5)

WELCOME!

ViSta - The Visual Statistics System - is ready to help you "Have fun seeing what your data seem to say".

ViSta's fun and playful approach to data analysis helps you see what your data seem to say, and to evaluate the truthfulness of what you think you've seen.

ViSta shows you multiple dynamic views of your data and provides you with numerous high interaction tools. The dynamic graphics and their interactive tools encourage you to play with your data, thereby augmenting your visual insight and increasing your understanding of the data.

So --- Welcome! And have fun seeing your data!

Forrest W. Young

ViSta - The Visual Statistics System
Wednesday, December 13, 2000 - 8:12 AM

Welcome to ViSta - The Visual Statistics System (2 of 5)

THANKS! :-))

Thanks for helping out with ViSta's Beta testing. Your efforts, and the efforts of all of the testers, help make ViSta a better system.

ViSta 6.1 is in final form. All features are completely implemented, except for the CrossTabs visualization. ViSta 6.1 is based on the ViSta Advisory Board's recommendations that I develop a ViSta core engine whose features will remain unchanged in the future.

The core engine is supposed to be LEAN and MEAN - Lean meaning that it is the smallest fully functional implementation that is useful for beginners - Mean meaning that it is fast and clean.

We will grow ViSta via Plugins and Addons which will add attitional capabilities for more advanced or specialized uses. The plugins, as well as the core engine, are available on the website. Currently, there are plugins for:
1) Multivariate Analysis
2) Homogeneity Analysis
3) Multilevel Analysis
4) Application Development
5) Log Linear Analysis (under development)
The multivariate plugin includes multivariate regression, principal component analysis, cluster analysis, multidimensional scaling, and correspondance analysis.

For developers, the application development plugin incudes a compiler, maker, a simple debugger, and a distribution maker. A CVS system has been setup for those who wish to do System Development. Please contact me directly for further information about using CVS.

There are two main features that are completely implemented for the first time in this release. These are:
1) ViSta-Excel connection. This feature, one of the most important new capabilities of ViSta 6, seems to be finally working reliably. Please test it.
2) Printing. It has been possible to print graphics with no problem in previous releases. However, this is the first time that the report, help and information windows can be printed.

A new panel for accessing help is also available.

I'd like comments and evaluations before I widely announce ViSta's availability. I'd especially like to know if it looks "ready to go": Are there important aspects that are not ready (by omission or comission)? Is it functional enough and small enough for the introductory social sciences undergraduate statistics student, or for the researcher with data but who had stats long ago, if ever? Is it fast? Stable? Mature? Smooth? Pretty? Easy to use? In short, how is the user experience?

A bug list is included with this release so that you can see if we know about the bugs you find. A menu item accesses it from the Options Menu. Another item you tell us of the bugs you found.

After this message you will see a series of messages. These messages are what the ViSta user will see the first time s/he runs ViSta. At the end of these is a registration. Please send it in, even if you have registered for a previous version.

I really appreciate the time and effort you and the other testers are taking to make ViSta a better statistical visualization system. Indeed, with your help ViSta can be the best freely available statistical visualization system --- perhaps the best, period. Thanks for helping out with the effort :-))

Forrest

ViSta - The Visual Statistics System
Wednesday, December 13, 2000 - 8:12 AM

Welcome to ViSta - The Visual Statistics System (3 of 5)

HELP!

We all need help, every now and then, so ViSta comes with a lot of help for when you need a little.

The HELP TOPICS item of the HELP menu displays the HELP PANEL that you see in the lower right corner of your screen (you can also type ? or HELP into the Listener window at the bottom of the desktop). The help panel lists the help topics and subtopics available to you.

This window is one help subtopic. As shown by the highlighting of the items in the panel, you are currently on the first topic, "Welcome to ViSta", and are reading its' subtopic HELP!

Each help topic includes several subtopics. For example, this "Welcome to ViSta" help topic includes subtopics that welcome you in, that describe the features of ViSta, that offer a poetic vision of statistics, and that provide you the opportunity to register.

Because of the importance of the first topic, you are being stepped through its subtopics. There are two more help topics that are "required reading"! The first is on "Using The DeskTop", and the second is on "Using The Menus". Please read these messages. They will help familiarize you with ViSta and its capabilities.

ViSta - The Visual Statistics System
Wednesday, December 13, 2000 - 8:06 AM

About Version 6 (4 of 5)

Welcome to ViSta 6!

ViSta 6 has evolved in a special way from ViSta 5. We haven't simply added new features for the sake of adding new features. Rather, we have evolved ViSta with one goal always foremost: Maximize the quality of the user's experience. With this guiding principle, we are pleased that ViSta is faster, smoother, and easier to use. It is also more flexible, expandable and capable.

FASTER and SMOOTHER

ViSta starts more quickly, loads data more quickly, and processes it more quickly. Since it is faster, it can analyze larger data. ViSta is also smoother. The visualizations and desktop work more smoothly, bringing you closer to your data, presenting a data analysis environment that is a more satisfying and relaxing, and bringing you a richer visual experience for a deeper understanding of your data.

EASIER TO USE

ViSta has a simplified user interface that involves only three types of windows: The DeskTop, SpreadPlot and Report windows. The DeskTop window has window-panes corresponding to the former workmap, datasheet, variables and observations windows. When you manipulate the DeskTop window all of its panes are manipulated accordingly, meaning there is only one window to manipulate rather than four. The same is true for SpreadPlot windows: They have panes corresponding to former windows: Manipulating one spreadplot window automatically manipulates all of the former windows in a coordinated way. Report windows continue to work in the same straight-forward way.

MORE FLEXIBLE

ViSta has PLUGINS and ADDONS that let you mold the data analysis methods and environment to fit your needs. PLUGINS add analysis methods to ViSta's basis methods, letting you tailor the selection of data analysis methods to those that are useful for you. ADDONS enhance the data analysis environment, letting you add new features as they become available. This permits ViSta's core system, and its unique selection of data visualization, exploration and description features, to remain stable and mature.

MORE CAPABLE

Just to identify 5 of the most important new capabilities:
1) ViSta 6 can be used as a visualization tool by Excel users, providing easy access to ViSta's dynamic visualizations.
2) ViSta 6 can accept and process data with missing values, and missing data imputation is provided.
2) ViSta 6 has a data manipulation language, allowing you to

manipulate, combine and create new variables.

4) ViSta has new visualizations, and all of the previous visualizations have been improved.

5) ViSta 6 has an expanded selection of analyses. With ViSta's core system you can do ANOVA, regression, frequency table analysis and significance testing. Plugins provide multivariate regression, principal components, cluster analysis, homogenity analysis, correspondence analysis and multidimensional scaling.

MORE EXPANDABLE

ViSta is an open system. Developers can write new plugins that introduce new data analysis capabilities, and new addons that add new features to the data analysis environment.

For more detailed information, select the help menu's NEW FEATURES item.

ViSta - The Visual Statistics System
Wednesday, December 13, 2000 - 8:06 AM

New Features in ViSta 6 (5 of 5)

ViSta 6 has numerous new features, including:

VISUALIZING EXCEL DATA	MISSING DATA FEATURES
BETTER DATA MANIPULATION	ADDITIONAL ANALYSES
NEW VISUALIZATIONS	MORE TRANSFORMATIONS
DATASHEET ENHANCEMENTS	NEW DATA FEATURES
USER INTERFACE IMPROVEMENTS	NEW MS-WINDOWS FEATURES

We hope you enjoy the new aspects of ViSta. If you have any suggestions for improvements, ideas for new features, bug reports, or code for plugins or addons you would like to contribute, please contact forrest@unc.edu.

VISUALIZING EXCEL SPREADSHEETS

You can use ViSta to visualize and analyze your Excel data. After a one-time configuration, each time you run Excel there will be a ViSta menu on the Excel menubar. You then simply select the portion of your spreadsheet that you want ViSta to visualize or analyze, and then select the menu item from the ViSta menu that is appropriate.

The items in the menu can be tailored to perform the specific types of visualizations and analyses that are appropriate for your data. All of ViSta's features can be accessed from Excel's ViSta menu.

NEW MISSING DATA FEATURES

Thanks to the efforts of Pedro Valero, of the University of Valencia, in Spain, ViSta can now process data with missing values. ViSta has methods for imputing values for the missing data, and has visualizations for inspecting the results of the imputations. In addition, most of the standard data processing methods can be used with data which have missing values.

PROCESSING MISSING DATA - Data with missing values, indicated by the symbol NIL, can be processed by ViSta 6. They can be input, manipulated, described and visualized in the same ways as data without missing values.

IMPUTING MISSING DATA - Three ways of imputing values for the missing data are now available. These include Maximum Liklihood, Casewise deletion and Pairwise deletion.

VISUALIZING IMPUTATIONS - Visualizations are provided to enable you to assess the results of the imputation methods, and to compare the imputed values obtained by the three methods.

BETTER DATA MANIPULATION

ViSta's data manipulation capabilities have been improved. The new capabilities, which are discussed in the HELP menu, include:

VIVA - ViSta's Interactive Variable Abacus, is an algebraic language for interactively manipulating variables.The algebrac-syntax is easy to use by most of those familiar with algebra. For example, you can type statements like:

```
[ gpa_normed = (gpa - mean(gpa)) / st_dev(gpa) ]
[ sattotal   = satmath + satverb ]
```

These statements create new variables (in this example, gpa_normed and sattotal) from pre-existing ones (gpa, satmath and satverb). Thus, ViVa provides a mechanism for manipulating variables which uses a familiar syntax.

VAR - If you prefer the power of Lisp (and don't mind the syntax), variables can also be created by ViSta's Lisp-based VAR function. The definition of the statement is:

```
(VAR VARNAME FORM)
```

where FORM is any Lisp form. For example, to perform the same calculations as those that are performed by the ViVa example just given, you would type:

```
(var gpa_normed = (/ (- gpa (mean gpa)) (st-dev gpa)))
(var sattotal (+ satmath satverb))
```

All variables in all data objects are available for use in ViVa, VAR, or in any other Lisp statement.

DATASET - The DATASET macro provides a simple way to create new data objects from variable objects. You only need to type

```
(DATASET DATANAME VAR1 VAR2)
```

to create a new data object containing the indicated variable objects. For example, you could create a data object containing the new variables calculated above by the statement:

```
(dataset scores gpa_normed sattotal gpa satmath satverb)
```

Taken together, the new ViVa language, and the new DATASET and VAR macros provide a simple but powerful way to manipulate variables.

NEW ANALYSES

ViSta's analysis capabilities, and the corresponding visualizations, have been expanded. These new capabilities include:

FREQUENCY ANALYSIS - Methods for analyzing and visualizing frequency data are now available in ViSta. The data may be formatted in several ways and may be converted between formats. The visualizations is based on Michael Friendly's work on Mosaic plots. Frequency analysis is available with ViSta's core features.

CLUSTER ANALYSIS - Cluster Analysis is available for the first time. Clustering may be done using a wide variety of methods and distance measures, based on work by Huh Moon Yul, Lee Kyungmi, and Jan deLeeuw. The visualization involves the familiar Dendogram and a newly developed Coloration Plot. Cluster analysis is available as a ViSta plugin.

HOMOGENEITY ANALYSIS - Jan deLeeuw's HOMALS analysis has been incorporated within ViSta. All of the power of homogeneity analysis is available, plus specially designed visualizations have been developed to communicate the results. Homogeneity analysis is available as a ViSta plugin.

NEW VISUALIZATIONS

ViSta's visualizations have been improved and new ones have been introduced. These new capabilities include:

T-TEST VISUALIZATION - The T-Test (formerly UniVar) visualization has been improved. It now is based on the appropriate ANOVA visualization rather than being the data visualization it was in the past.

REGRESSION VISUALIZATIONS - Four new regression visualizations have been developed, one for each of the four kinds of regression analysis that are available (simple, OLS, Robust and Monotone).

VISUAL CROSSTABS - An "awesome" (to quote the response of several who have seen it) highly interactive visualization for crosstabs data is now provided.

CATEGORICAL DATA VISUALIZATION - Categorical data now have their own data visualization, using ViSta's new mosaic plots.

NEW TRANSFORMATIONS

VISUAL BOX-COX
VISUAL FOLDED POWER TRANSFORMATION
DUMMY VARIABLE TRANSFORMATION

Pedro Valero has added these three new transformations to the transformation menu.

USER DEFINED TRANSFORMATIONS - You can use the new USER DEFINED item of the TRANSFORM menu to write your own transformations using ViVa or Lisp. All of your variables are accessible and can be modified. New variables can be computed from old ones.

DATASHEET ENHANCEMENTS

EMPTY DATASHEET - An empty datasheet is presented at the beginning of a session with ViSta, increasing convienience of using ViSta when you wish to enter new data.

ALWAYS EDITABLE - Datasheets are now always editable, and the data object is now always updated with every keystroke, thereby eliminating confusing situations in which ViSta could not perform analyses (since the data object had not been updated) and simplifying the process of saving data (the confusing dialog box is gone).

INTERFACE CHANGES - Datasheets have a new toolbar and respond to right-button clicks.

USER INTERFACE IMPROVEMENTS

In addition to the major enhancements and additions reviewed above, numerous improvements have been made to the ViSta's user interface. These include the following:

SPREADPLOTS - SpreadPlots can be reopened to their previous state.

Several SpreadPlots can be open simultaneously. Multiple SpreadPlots can be defined (and can be open) for a single data or model object.

GRAPHICS - New histograms with frequency or probability axis, improved dynamic binning, explicit bin mid-points, better bin cut-points and hollow histograms (based on algorithms developed by Jan de Leeuw and Jason Bond). Dynamic, highly interactive bar graphs (side-by-side and stacked). Frequency polygon plots with dynamic, highly interactive binning (also thanks to Jan and Jason). Enhanced mosaic plots, now with residual-based coloring and dynamic highlighting. New quantile-based contour plots. New smoothers and distribution curvers for several plots. Improved labeling of many plots. Many of the dynamic graphics features have been made faster and smoother.

WORKMAP - Icons can be deleted (finally!). Icons have been introduced for spreadplots and reports. The redesigned data icons visually cue the new data types. The tool bar has user-definable analysis buttons (and a new appearance). Object names can be edited

NEW DATA FEATURES

DATA TYPES - ViSta now has seven data-types: Multivariate (including uni-variate and bi-variate); categorical; classification; frequency tables; frequency classifications; matrices and missing data. Intelligence has been added in selecting the appropriate kinds of analyses for specific data types.

EFFICIENCY - The efficiency of Data loading and of DataSheet operations has been greatly improved, permitting much larger datafiles (100000 observations on 100 variables is now practical, although visualization is limited).

CONVERSION - Data can be exported as standard text. Imported data can include variable names and observation labels. More datatype conversions supported.

NEW MS-WINDOWS FEATURES:

Window maximize/minimize buttons work. Default button behavior improved in dialogs. Main window position and size remembered between sessions. Cursor keys work. OPEN FILE and EDIT FILE menu items are now available. Installation has been simplified and standardized. The networked Windows NT installation is easier and more flexible.

ViSta - The Visual Statistics System
Saturday, December 9, 2000 - 6:21 AM

Help: DeskTop (1 of 7)

ABOUT VISTA'S DESKTOP

ViSta's DeskTop window has four panes --- sections of the window with their own title and scroll bars. By default, you can see all panes at the same time. You can use the various MAXIMIZE items of the DESKTOP menu to focus exclusively on one specific pane. The RESTORE item of the DESKTOP menu restore your view of all four panes.

The panes are described briefly below. The HELP menu and the HELP buttons in the panes provide more complete help information.

WORKMAP
The WorkMap window-pane maps the steps you take during a data analysis session. The map is constructed and displayed as your analysis progresses. At first, before you start analyzing your data, there is no map. As you step through your data analysis, icons representing data analysis steps are added to the WorkMap. The icons are usually connected to previous icons by lines that show the flow of your data analysis steps.

DATASHEET
The Datasheet window-pane displays your data, and lets you enter new data or edit existing data.

SELECTOR
The Selector window-pane displays lists of the observations and variables in the currently active data object (when there is one). You can select items in these lists to form subsets of the data.

LISTENER
The Listener pane displays messages about the system, and lets you type commands in ViSta's underlying languages (see the CALCULATOR item of the HELP menu for more information).

Using The WorkMap (2 of 7)

ABOUT VISTA'S WORKMAP

The WorkMap is the heart of ViSta's data analysis and visualization environment. It provides an up-to-date picture of your data analysis session, and, with the help of the Toolbar, is the control center where you interact with ViSta to understand your data. The WorkMap is where you create analyses and visualizations of your data.

The WorkMap records the steps you take during a data analysis session. The map is more than just an aid for remembering what you've done: You can interact with the map to re-visit previous steps in your analysis, and, when you wish, to analyze your data in new ways.

The WorkMap grows as your analysis progresses. At first, before you start analyzing your data, there is no map. As you step through your data analysis, icons representing the statistical objects created by your data analysis steps are added to the WorkMap. The icons are usually connected to previous icons by lines that show the flow of your data analysis steps.

The icons represent the statistical objects which are the focus of your session with ViSta. These objects include:

> DATA OBJECTS, which are the data you are analyzing and data derived from your data;
> ANALYSIS OBJECTS, which are mathematical programs for extracting information from data objects;
> MODEL OBJECTS, which are mathematical models of your data derived by the analysis objects;
> VISUALIZATION OBJECTS, which are pictures of your data or of the models of your data; and
> REPORT OBJECTS, which are verbal and/or tabular summaries of the data or models.

You obtain information about a statistical object by clicking on its icon. Clicking on an icon in various ways has various effects. These effects are described below:

> SINGLE CLICK: A single click on an unselected icon selects the icon, changing its highlighting and making the icon's data the current data. The data are displayed in the DeskTop's DataSheet pane, and the observations and variables are shown in the Selector pane.

> DOUBLE CLICK: A double click on the main body of an icon opens the icon, revealing its' information (see below). Double clicking the bar at the top of the icon presents you with summary information about the statistical object. Double clicking on the icon's name presents a dialog box for changing the icon's name.
  OPENING DATA reveals the data's datasheet.
  OPENING MODELS reveals the model's data objects.
  OPENING ANALYSES reveals the analysis' parameters.
  OPENING VISUALIZATIONS reveals the visualization's graphs.
  OPENING REPORTS reveals the report's tables and/or text.

> RIGHT CLICK: A right-click anywhere on an icon displays a menu of

actions you can take that are appropriate to the statistical object represented by the icon.

> DRAG: You can drag icons to a new location by placing your cursor on an icon, pressing your mouse button down, and dragging to a new location while holding the button down.

> CTRL-DRAG: You can also drag icon-trees (an icon and all the icons attached below it) by holding down the CTRL key before dragging.

Using The DeskTop Options (3 of 7)

DESKTOP OPTIONS:

Three items of the OPTIONS menu effect the nature of the desktop. These are the DESKTOP LAYOUT, DEKSTOP COLORS, and WORKMAP OPTIONS items. We discuss these items and their effects, here.

MODIFYING DESKTOP LAYOUT:

You can change the appearance of the DeskTop with the OPTION menu's DESKTOP LAYOUT item. The item brings up a menu which provides the following choices:

WORKMAP PROPORTION OF WORKMAP+DATASHEET HEIGHT: Controls the height of the WorkMap (and Selector) relative to the height of the DataSheet.

NUMBER OF LISTENER LINES: Controls how many lines of the listener are visible. The WINDOW menu's MAXIMIZE LISTENER overrides this specification.

TITLE BARS determines whether the window-panes show their title bar.

THIN BORDERS determines the type of border between window-panes.

RESTART NOW restarts ViSta after you close the dialog, in case you want changes to take effect right away.

MODIFYING DESKTOP COLORS

Use the DESKTOP COLORS item of the OPTIONS menu to change the colors of various DeskTop features.

MODIFYING ICON APPEARANCE AND LAYOUT:

You can change the appearance and layout of the icons on the workmap with the WORKMAP ICONS item of the DESKTOP menu. The item brings up a menu which provides the following choices:

WHEN TO SHOW SMALL GRAPH/STATS ICONS: Controls when the small report and visualization icons are shown attached to the side of the main icon.

TITLES HAVE WHITE BACKGROUND? When checked, the titles are given a white background, which helps distinguish the title from the background of the desktop.

TITLES ARE SHORT?: When checked, only the first 8 or 9 characters of the title are shown. This can improve legibility by preventing titles from overlapping. When not checked the entire title is shown.

TITLES INCLUDE DATA VERSION NUMBERS. When checked, the data object's version number appears in the data menu and workmap. Version numbers are used to identify data objects with otherwise identical names.

ZIG-ZAG ICON LAYOUT: When checked, icons are vertically offset to prevent title overlap. Otherwise no offset is used.

Using The WorkMap ToolBar (4 of 7)

ABOUT VISTA'S WORKMAP TOOLBAR

The ToolBar contains buttons which provide instant access to ViSta's analyses, visualizations and reports. The left three buttons provide access to Help and to a statistical (data or model) object's report and visualization. The right-hand group of buttons provide quick access to various ANALYSIS possibilities. These buttons correspond to items of the ANALYSIS menus.

ViSta can decide to a limited extent what analyses are appropriate for the kind of data you are using. This decision is reflected in the changing appearance of the buttons (and of the corresponding menu items in the ANALYSIS menu). An active (colorful) button is judged potentially appropriate. An inactive (gray) button is judged definitely inappropriate. The ANALYSIS menu items change accordingly.

Clicking an activated (not gray) tool button carries out the action of the button. For the left three buttons, these actions are Help, Reports and Visualizations. For the rest of the buttons, these actions are specific analyses.

Right-clicking the toolbar gives you a menu which lets you redefine the number of buttons and the button actions.

ViSta - The Visual Statistics System
Sunday, December 10, 2000 - 10:54 AM

Help: DataSheet (5 of 7)

ABOUT VISTA'S DATASHEETS

Datasheets display your data and let you edit your data. Datasheets can be used to create a brand new data object, to change the data in an already existing data object, or to add new data to an already existing data object.

Datasheets support the standard editing functions. Click on a cell to edit it. Type the desired information. Use the delete key to delete what you have typed. Use the cursor keys or the return, home, end or tab keys to move to another cell.

Datasheets are not spreadsheets: They do not let you enter mathematical formulas in the cells. To do this, use a spreadsheet such as Excel. When you have the desired spreadsheet you can transfer it to ViSta as a datasheet, taking advantage of ViSta's advanced and extensive vizualization and analysis features. When you make the transfer from Excel to ViSta, Excel's numerical and textual information is transferred as is, as are the values resulting from the Excel's formulas. The formulas themselves are not transferred.

CONTENTS OF THE DATASHEET:
The left column of cells contains observation labels. The top row of cells contains variable names. These may contain any information that you wish. It is recommended that each label and name be unique.

The second row of cells contains variable types. ViSta recognizes three variable types: Numeric, Ordinal and Category (capitalization is ignored). Any other information typed in these cells will cause errors.

The remaining cells of the datasheet contain the data. You may type any information you wish. However, Numeric and Ordinal variables assume that their cells contain numbers. If they do not an error message will be displayed when you try to save the data. Category variables treat the information that you type as strings that do not have to be numbers. The datasheet can contain missing data. Missing values are entered as blank cells, and are displayed in the datasheet as blanks.

REFORMATING THE DATASHEET:
You expand the datasheet (add rows, columns or matrices) by clicking on special cells in the datasheet. To add a new observation (row) to multivariate data, click on the "New Obs" cell located below the left-hand column. To add a new variable (column) to the data, click on the "New Var" cell located to the right of the top row. You expand matrix data in a similar fashion. You can also use the EXPAND button (or menu item) to add several rows or columns simultaneously. You can control the width of columns and the number of decimal places shown by using the FORMAT button on the button bar,or appropriate items of the DESKTOP menu.

SAVING THE DATA:
You may use the button bar's SAVE button,or the FILE menu's SAVE DATA

menu item to save the data. When you save the data, the current data object will be updated, unless it is attached to another icon. If so, the old data object will be left unchanged, and a new one will be created from the information in the datasheet.

ViSta - The Visual Statistics System
Sunday, December 10, 2000 - 10:54 AM

Using The Selector (6 of 7)

The SELECTOR displays lists of the observations and variables in the currently active data object (when there is one). You can select items in these lists to form a subset of ACTIVE observations and variables.

ACTIVE variables and observations are those which are highlighted, or if none are highlighted, those which are listed.

Only the ACTIVE variables are used in ViSta's analyses and visualizations.

You can create a new data object containing only the ACTIVE observations and variables with the DATA menu's CREATE DATA menu item.

You make a SELECTOR item ACTIVE by clicking it. You can select several items by dragging your cursor over them. You can add items to a selection by CTRL-clicking, or CTRL-dragging.

You form a selection with the buttons at the top of the SELECTOR. These buttons let you put data items IN the selection or take them OUT of the selection. You can also DROP a selection or RESET all data items to the selection.

Using The Listener (7 of 7)

The Listener Window displays messages about ViSta's operation and provides a place for typing statements in the underlying Lisp language, and in ViSta's ViVa language.

There are two Listener Windows: The DeskTop Listener, which is in the main desktop window, and the XLispStat Listener, which is in the XLisp-Stat window. You need to use the WINDOW menu's XLISPSTAT WINDOW item to see the XLispStat window. Only one of these is active at a time. Both work identically.

Several items of the WINDOW menu control aspects of the DeskTop Listener Window. The MAXIMIZE LISTENER menu item lets you maximize its size, while HIDE LISTENER hides it from view. The RESTORE LAYOUT item changes the desktop back to the basic layout. The DESKTOP LAYOUT item lets you change the number of lines shown in the DeskTop Listener.

To save the information that you see in the Listener, use the OPTIONS Menu's RECORD LISTENER item. This item is a toggle which turns recording on and off. All information that appears in the window while the item is checkmarked (e.g., AFTER you use the item the first time and before you use it again) will be recorded in the file you specify in the file dialog.

For more information about using the Listener for programming, see Luke Tierney's book on Lisp-Stat.

ViSta - The Visual Statistics System
Sunday, December 10, 2000 - 10:54 PM

Using the Menus (1 of 4)

ABOUT VISTA'S MENUS:

The menus are arranged in order, from left to right, in a way which reflects the order of the steps taken during data analysis.

First you use the FILE menu to get data into ViSta. It has items to create NEW DATA; to OPEN DATA for analysis by ViSta; to SIMULATE DATA according to statistical models; and to IMPORT DATA from text files. It also has items for exporting data and for inputing and saving text files.

The EDIT menu is not directly focused on data analysis, but provides the usual copying, pasting, etc. You can delete objects from the DeskTop, and you can copy any window to the clipboard. The remaining items can only be used with the Listener Window.

The DATA, TRANSFORM, ANALYZE and MODEL items are the main data analysis menus. They are also the main four steps in understanding data, and their arrangement order reflects the order in which these steps occurs when looking at your data:
1) Use the DATA menu "to see what the data seem to say" and to manipulate the data;
2) Use the TRANSFORM menu to transform the data (whether this is necessary depends on "what you see" in step 1);
3) Use the ANALYZE menu to use an analysis of your data to produce a model of what your "data seem to say"; and
4) Use the MODEL menu to look at the model resulting from the analysis (again, "to see what the data seem to say", but this time we look at the view the model provides).
A typical data analysis is a cycle repeating the above steps.

The OPTIONS, HELP and WINDOW menus, respectively, allow you to change ViSta's data analysis environment, provide help about using ViSta, and provide access to ViSta's other major windows. Each of these menus has its own separate help information which you can access through the HELP menu.

Options Menu Help (2 of 4)

HELP FOR THE OPTIONS MENU

The OPTIONS menu changes the way that ViSta works. The changes you make are saved and will be in effect when you use ViSta again. Details are given in this help message.

DESKTOP LAYOUT:
Use this to change the relative proportions of the various window-panes of the desktop, and to remove or add title bars to the window-panes.

DESKTOP COLORS:
Changes the colors of windows, of tool and button bars, and of icons and graph elements. You can set the maximum number of colors to correspond with your hardware's capabilities.

WORKMAP ICONS:
Changes appearance of the icons that appear on the workmap.

FLOATING WORKBAR:
Displays a floating workbar of menus and tools.

TOOLBAR BUTTONS:
You can change the function (and number) of the buttons on the toolbar at the top of the workmap.

CHANGE FONT:
Initially, fonts and font sizes are set so that they look right with ViSta. You can use this item to change to other fonts or sizes, however, proportional width fonts will never look right, and some fixed width fonts also do not look right.

RUN EXCEL
This item runs Excel, adding a VISTA menu to Excel. The menu has items which can be configured by the experienced ViSta user. See the help item for these features.

SHOW CLOCK:
Allows you to turn the morphing vista logo clock on and off. When the clock appears ViSta disappears, and can be made to reappear when the clock window is clicked.

RECORD LISTENER:
Provides a way of recording the information that is written to the listener. This provides a partial, but not complete, mechanism for journaling your work.

REFRESH SYSTEM:
Provides a way of refreshing the system after errors so that too many errors will not accumulate, causing "Stack Overflow" problems. The number of error levels (roughly the same as the number of errors since the system was refreshed) is indicated by a number in front of the Listener Window's prompt. Thus 2> indicates, roughly, two errors since refreshing. It is safest to use this item whenever the Listener window's prompt has a number in front of it.

ViSta - The Visual Statistics System
Wednesday, December 13, 2000 - 8:09 AM

Window Menu Help (3 of 4)

HELP FOR THE WINDOW MENU

The WINDOW menu lets you display ViSta's windows and lets you configure ViSta's desktop window. The changes you make are saved and will be in effect when you use ViSta again. Details are given in this help message.

DESKTOP WINDOW
XLISPSTAT WINDOW
SPREADPLOT WINDOW
REPORT WINDOW
These items display the specified window

MAXIMIZE WORKMAP
MAXIMIZE DATASHEET
MAXIMIZE LISTENER
RESTORE LAYOUT
The MAXIMIZE items effect the window panes of the desktop window by maximizing the specified pane to occupy the entire height of the DeskTop window. RESTORE LAYOUT item restores maximized panes to their size prior to maximization.

HIDE/SHOW SELECTOR
HIDE/SHOW LISTENER
These items hide or show their respective window panes.

MAXIMIZE DESKTOP
RESTORE DESKTOP
REFRESH DESKTOP
These items effect the desktop window. REFRESH DESKTOP item refreshes the window. This item is useful if the contents of the window do not appear to be drawn correctly.

MENU ITEM HELP

The HELP menu's MENU ITEM HELP item turns menu help on and off. It changes the function of the menu items to give you help about what they do rather than actually doing what they normally do. This applies to the items of the FILE, DATA, TRANSFORM, ANALYZE, MODEL and OPTIONS menus.

When MENU ITEM HELP is OFF, the menu items carry out the actions they normally perform.

When MENU ITEM HELP is ON, menu items provide help about what they normally do, but don't perform the action.

You can tell if MENU ITEM HELP is ON or OFF by looking at the help item. When you pull down the menu, you will notice the MENU ITEM HELP item. Look to see if it is check-marked (with a check-mark to the left of the item). If it is check-marked, then menu item help is ON, showing that the menus are ready to provide help. When the item is not check-marked the menu system works normally.

When you no longer wish to see help information you may use the MENU ITEM HELP item to toggle menu item help off. The menus will then work normally.

Excel and ViSta (1 of 2)

ABOUT USING EXCEL WITH VISTA

You can use ViSta to visualize and analyze your Excel data. The process involves using Excel to transfer you data to ViSta. ViSta will then visualize or analyze your data according to instructions sent with the data. The process involves the following 5 steps:

1) RUN VISTA
RUN VISTA before you run Excel. If you run Excel first you may not be able to connect with ViSta (as explained below).

2) RUN EXCEL
Use the RUN EXCEL item of ViSta's OPTIONS menu to run Excel. You will see Excel. After a short delay you will also see the ViSta menu and a minimized spreadsheet named EXCELVISTA. Do not remove the EXCELVISTA spreadsheet. If you do, the ViSta menu will be removed too.

3) ENTER DATA
You can use the EXCELVISTA speadsheet to enter new data or you can retrieve previously defined data as you normally would. The data must satisfy requirements that are given below.

4) SELECT DATA
Select the portion of your spreadsheet that contains the data you wish to use with ViSta.

5) CHOOSE A VISTA MENU ITEM
Choose a menu item from Excel's ViSta menu. After a moment ViSta will show you the results of your choice.

Now you can use ViSta to look at your data. You can switch back and forth between ViSta and Excel at any time. You can use other items of Excel's ViSta menu, and you can get new data and use the menu items to look at it with ViSta. You cannot, however, quit and restart ViSta without also quitting Excel, as this may make reconnection impossible. You can quit both programs and then use ViSta to get Excel going again.

There are the following details to consider:

DATA REQUIREMENTS
a) The data must be in a rectangular array of contiguous cells.
b) The top row of cells must contain variable NAMES. The NAMES can be anything you wish, but cannot contain single or double quote marks. NAMES are used by ViSta to identify columns of data.
c) The second row must contain variable TYPES. The variable TYPE can only be NUMERIC or CATEGORY (upper or lower case, no quote marks).
d) The left column must contain observation labels. Labels can be any information you desire. They are used by ViSta to identify the rows of data.
e) The first cell in the NAMES row and the first cell in the LABELS row are ignored.

MACROS: During the process of using ViSta with Excel, you may be warned that a speadsheet contains a macro, or you may be told that you are being prevented from loading a spreadsheet because it contains a macro. You should not be worried, as the instructions for Excel to work with ViSta are contained in the macro. You may need (or wish) to change your security settings so that the macro can be loaded more easily. When all is right, the macro installs the ViSta menu on Excel's menubar.

EXCEL FIRST: If you use Excel's ViSta menu and ViSta isn't running, Excel will

automatically try to run ViSta. However, it is possible that the "PLEASE TRY LATER" error message will appear, and that no matter how long you wait you will never be able to run ViSta. This situation arrises when ViSta is running, but is hidden from view so that you don't know it is running. If the error message appears, you can quit Excel, and use the LispEditor to evaluate an (exit) function to terminate the invisible ViSta process. Note that this process itself generates an error message! If it says "Cannot Connect with XLispStat" you have removed all invisible occurances of ViSta. If it says something else, you should then re-evaluate the (exit) function. Continue until you get the "Cannot Connect with XLispStat" message.

Excel's ViSta Menu (2 of 2)

CHANGING THE ITEMS OF EXCEL'S VISTA MENU

This help tells you how to change the items in Excel's ViSta menu. You can remove the ones that are there, add new ones, etc. The actions of the menu items can be any ViSta Lisp script. Thus, you can have memu items that do any type of visualization/analysis that ViSta Lisp is capable of.

The help information is in two parts: The first is about what ViSta does when an Excel ViSta menu item is used. The second is about what you have to do to modify the information sent along with the data by Excel to ViSta.

WHAT VISTA IS DOING

The nature of the connection between Excel and ViSta is up to you. You can change the items of the ViSta menu in Excel so that they perform actions controlled by ViSta scripts that you write.

The process rests on knowing what ViSta is doing at the moment that it uses the information in the scripts. That moment might either be while ViSta is being started up by Excel, or it might be while ViSta is already running, having been started somehow in the past.

When ViSta is first run, it is loaded into memory and gets to a point in time where it determines whether it is being started by Excel or in some other way. At this point in time ViSta is in "startup mode": The opening splash window (containing the spinning logo) has just closed and ViSta is ready to finish starting.

If ViSta was started long ago ("long ago" according to the computer's view of time, anyway) it will be just sitting around doing nothing in particular. It won't be in startup mode, and the splash window will have been long closed.

In either case, ViSta processes Excel's data with the Lisp function
    (process-dde-data)
which gets the data into ViSta, taking care of all the messy details. The function also associates the data with a script which performs the visualization/analysis.

Because Excel may send ViSta data while ViSta is in the middle of its startup proces, the following functions are available to you for controlling the remainder of the startup process. Furthermore, they do the right thing if ViSta has finished starting.

(startup-show-vista)
    If ViSta is in startup mode, this function opens ViSta with the normal (non-excel) sequence, otherwise the function brings the DeskTop window to the front

(startup-remove-logo)
    If ViSta is in startup mode, this function removes the splash (spinning logo) window, otherwise does nothing

(startup-exit-on-close)
    This function makes the close box of the window created by the preceeding script statement act as an exit box.

WHAT YOU MUST DO

Take the following steps to customize the menu items in Excel's ViSta menu:

1. Prepare the Scripts

Write the scripts to be used by the Excel menu items. Look at the scripts in the startup\excel directory to see how they were prepared.

2: Save the Scripts

Each script must be saved as a file in the startup\excel directory. The filename must match the name in the excelmenus.txt file (see next step).

3. Change the Menus:

Look at the existing excelmenus.txt file to see how the information in it connects the menus you see in Excel with the scripts in the startup/excel folder. Then edit the file, entering the new menu name followed by the lisp script associated with it.

Using the DataSheet (1 of 3)

ABOUT VISTA'S DATASHEETS

Datasheets display your data and let you edit your data. Datasheets can be used to create a brand new data object, to change the data in an already existing data object, or to add new data to an already existing data object.

Datasheets support the standard editing functions. Click on a cell to edit it. Type the desired information. Use the delete key to delete what you have typed. Use the cursor keys or the return, home, end or tab keys to move to another cell.

Datasheets are not spreadsheets: They do not let you enter mathematical formulas in the cells. To do this, use a spreadsheet such as Excel. When you have the desired spreadsheet you can transfer it to ViSta as a datasheet, taking advantage of ViSta's advanced and extensive vizualization and analysis features. When you make the transfer from Excel to ViSta, Excel's numerical and textual information is transferred as is, as are the values resulting from the Excel's formulas. The formulas themselves are not transferred.

CONTENTS OF THE DATASHEET:
The left column of cells contains observation labels. The top row of cells contains variable names. These may contain any information that you wish. It is recommended that each label and name be unique.

The second row of cells contains variable types. ViSta recognizes three variable types: Numeric, Ordinal and Category (capitalization is ignored). Any other information typed in these cells will cause errors.

The remaining cells of the datasheet contain the data. You may type any information you wish. However, Numeric and Ordinal variables assume that their cells contain numbers. If they do not an error message will be displayed when you try to save the data. Category variables treat the information that you type as strings that do not have to be numbers. The datasheet can contain missing data. Missing values are entered as blank cells, and are displayed in the datasheet as blanks.

REFORMATING THE DATASHEET:
You expand the datasheet (add rows, columns or matrices) by clicking on special cells in the datasheet. To add a new observation (row) to multivariate data, click on the "New Obs" cell located below the left-hand column. To add a new variable (column) to the data, click on the "New Var" cell located to the right of the top row. You expand matrix data in a similar fashion. You can also use the EXPAND button (or menu item) to add several rows or columns simultaneously. You can control the width of columns and the number of decimal places shown by using the FORMAT button on the button bar,or appropriate items of the DESKTOP menu.

SAVING THE DATA:
You may use the button bar's SAVE button,or the FILE menu's SAVE DATA

menu item to save the data. When you save the data, the current data object will be updated, unless it is attached to another icon. If so, the old data object will be left unchanged, and a new one will be created from the information in the datasheet.

ViSta - The Visual Statistics System
Sunday, December 10, 2000 - 10:56 PM

Importing Data (2 of 3)

IMPORT DATA opens a file and attempts to import data from it. If the information in the file is formatted as explained below, ViSta will load the data and display it as a datasheet.

A dialog box will ask for the name of the data, whether the first line in the file contains variable names, whether the second line contains variable types, whether the first column contains observation labels, whether you want to see the data sheet, and whether you need to check missing data.

When the first line of the file specifies variable names, there must be one name for each variable, each one separated by white space. If the first variable name is LABELS, or if the "First Column is Labels" dialog item is checked, then the first "variable" will be used as observation labels. Variable names are unquoted strings. Optionally, they may be inside double quotes, permitting white space inside the name.

If the first line specifies variable names, then the second one can specify variable types, if desired. This can greatly speed up importation for large data. Variable types must be one of the following double quoted strings: "label" "labels" "category" "numeric". Case is ignored. If the second line does not specify variable types, then a variable that has one or more non-numeric values is treated as a category variable, otherwise variables are assumed to be numeric.

When the first line does not contain variable names it must contain the first data line. The line must have one data value for each variable.

Choosing to not see the datasheet can greatly improve the speed for importing large data.

If there are no missing values, or if missing values are all coded by the symbol NIL, and none are coded by the string "NIL", then you do not need to check missing data, a process which can be time-consuming.

Data values must be separated by white space. If variable names are not specified, then the number of data values in the first record determines the number of variables in the data object. Succeeding records must have the same number of data values.

Data values may be numeric or non-numeric. Non-numeric values may optionally be in double quotes. Double quotes permit entry of spaces, and preserve character case (unquoted non-numeric values are converted to upper case). Missing values must be coded as nil or "nil".

After you import data you will see a datasheet of your data. You may use this datasheet to browse through your data and to edit your data.

When you are finished with the datasheet you should close it. You will then see that other ViSta windows have changed: The WorkMap will have a new Data Icon. The name of the Data Icon will correspond to the name you entered into the dialog box.

If you are using guidemaps, the GET YOUR DATA guidemap will be replaced by the DATA ANALYSIS guidemap.

ViSta - The Visual Statistics System
Sunday, December 10, 2000 - 10:56 PM

Simulating Data (3 of 3)

SIMULATE DATA lets you generate new data by simulating the process of sampling from a population. It also includes a visualization that demonstrates the central limit theorem.

When you use SIMULATE DATA, you will see a dialog box that asks you to select a distribution. This corresponds to the theoretical population distribution from which samples will be drawn.

The dialog box also asks for sample size and number of samples, and whether or not you wish to visualize the sampling process. For certain distributions, an additional dialog box will ask for information needed to completely specify the sampling process.

The visualization shows the shape of the population and sample distributions as well as the distribution of sample means and standard deviations. If you ask for more than one sample, the last sample distribution is displayed along with all sample means and standard deviations. The visualization gives you the choice of generating additional samples and seeing the updated distributions of means and variances. Each time you click on the visualization's "New Sample" button ViSta generates "n" additional samples, where "n" is the number of samples specified in the dialog box.

The simulation process creates a new multivariate data object. It is represented by an icon on the workmap.

About The Data System (1 of 6)

ABOUT VARIABLES AND DATA

An important aspect of statistical data analysis and visualization involves creating new variables, calculating new variables from existing ones, and modifying the values of existing variables. Such new variables become the basis of new data objects, which in turn can be used for data analysis and visualization.

This help file presents an overview of variables and data objects, including information about ViSta's lists of variable and data object names; about the difference between short names and long-names; about how you use the lists of names to refer to variable and data objects; about how you create new variable and data objects; and about variable types and data types.

ABOUT SYMBOLS

You manipulate variables by typing simple arithmetic or algebraic statements in the listener window. These statements involve symbols for the mathematical operations and symbols for the variables. The mathematical operation symbols are the familiar ones you have always used: + for addition, / for division, etc. The variable symbols are the variable names you supplied when you created the variables.

VARIABLE NAMES - SHORT AND LONG

Throughout ViSta, variables and data objects are referred to by names that you supply. However, there is always the possibility that a name might be repeated. If it is, the name does not unambiguously identify the variable or data object. To avoid this problem, ViSta constructs and uses "long-names" which are unambiguous. In contrast, the "short-names" are the names you supplied, which can be ambiguous.

The long-name of a data object is the name you supply followed by # (the number sign) and a number supplied by ViSta. The number is 1 if the name is unique, 2 if there is already another data object with the same name, etc.

The long-name of a variable is constructed from the name you gave the variable and the name of the data object that contains the variable. Specifically, a variable's long-name is the name of the data object (including the number) followed by a dot and the name you gave to the variable;

    dataname#n.varname

For example, if the data object MYDATA contains a variable named GPA, then the variable's long-name is MYDATA#1.GPA when there is no other data object named MYDATA.

Variables do not have to be in data objects, but may be "free-standing" variables. When you first create and manipulate new variables they are free-standing. Free-standing variables have long-names which are the same as their given name since they are not in any data object. Note that there is the possibility of duplicate names.

There is always a "current" data object. This is the data object that is

the current focus of data analysis and visualization. It is represented on the workmap by the highlighted data icon.

REFERING TO VARIABLES

In order to manipulate a variable, you must be able to specify which variable you wish to manipulate. A variable or data object can be referred to by its name, including either its' short-name or long-name. If you use a short-name, the most recently created variable with that name will be the one referred to.

ViSta maintains information about the data and variable objects that have been defined during the session. The information can be referred to by various names. In addition to being able to refer to a data object by its' name (with or without the #n suffix), you can refer to the information by various symbols, all beginning with $. The symbols include:

NAME	INFORMATION
$	the current data object
$vars	list of variables in the current data
$data	list of all data objects
$all-vars	list of all variables
$data-vars	list of all data object variables
$free-vars	list of all free variables
$NAME-vars	list of all variables in data object NAME

By typing one of these symbols in the listener you can see which data and variable objects have been defined. New variables are usually constructed from specific variables on these lists. In addition, new variables may involve calculations performed on variables on these lists.

CREATING VARIABLES WITH VIVA AND VAR

ViSta provides two ways of creating new variable objects, known as ViVa and Var.

ViVa is ViSta's Variable language, and VAR is a function for creating variables. Each creates new variables which can be further manipulated with ViVa and Var, or can be made into data objects, using the DATASET function. The data object can then be analyzed and visualized by ViSta.

ViVa calculates variables using statements like ordinary arithmetic or algebraic statements. VAR calculates new variables using expressions in the Lisp language.

The variables created by ViVa and Var are called "free" variables because they are not contained in any data object.

ViVa and VAR are described in the CREATING VARIABLES help item.

THE DATASET FUNCTION

The "free" variable objects created by ViVa and VAR must be placed in a data object so that they can be analyzed and visualized. The DATASET function creates a new data object from a group of variables. It is described in the ABOUT MAKING DATA help file.

TYPES OF VARIABLE AND DATA OBJECTS

Variables have a property known as their "variable type", a property which determines what kinds of calculations can be done on their values and what kinds of statistics and visualizations are appropriate. The types recognized by ViSta are "numeric", "ordinal" and "category".

> Numeric variables are regular numbers, supporting ordinary arithmetic and the statistics and visualizations based on arithmetic.
> Ordinal variables are those whose values only specify order, not number. Such variables are seldomly used in ViSta.
> Category variables are those whose values only specify category membership. These variables are used in appropriate places in ViSta for determining the kinds of statistics and visualizations that can be computed.

By default, ViSta assumes variables are numeric. The VAR function's :TYPES keyword allows you to specify non-numeric variables. ViVa variables are always numeric.
f
DATA TYPES

Data objects have a property known as their data type. The data types recognized by ViSta are:

> UNIVARIATE data have a single numeric or ordinal variable.
> BIVARIATE data have two numeric or ordinal variables.
> MULTIVARIATE data have three variables and are data which are not one of the other data types given below.
> CATEGORY data have one or more CATEGORY variables and no NUMERIC or ORDINAL variables. The N category variables define an n-way classification.
> CLASSIFICATION data have one NUMERIC variable and one or more CATEGORY variables. The N category variables define an n-way classification. The numeric variable specifies an observation for a given classification.
> FREQUENCY CLASSIFICATION data are classification data whose numeric variable specifies frequencies. The N category variables define an n-way classification, with the numeric variable specifying the co-occurance frequency of a specific combination of categories.
> FREQUENCY TABLE data are data whose observations and variables are used to form the rows and columns of a two-way table. That is, the data are a two-way cross tabulation of the co-occurance frequency formed from the observations and variables. The data elements must be frequencies.
> MATRIX data are data whose observations and variables refer to the same things. These things are used to form a square, usually symmetric matrix with the same number of rows and columns, the rows and columns identifying the same things. The values might be correlations, covariances, distances, etc. Optionally, there can be more than one matrix in a given data object. All matrices must have rows and columns identifying the same things.
> MISSING data are data which have one or more missing values (coded as nil). These data must have their missing values imputed before analysis or visualization is possible.

Manipulating Variables (2 of 6)

MANIPULATING VARIABLES

A variable is a specific type of information (either numeric or non-numeric) that is observed many times, and which can vary from one observation to the next. Data is a collection of variables, and is the information that you wish to visualize and analyze with ViSta.

Before visualizing or analyzing your data, and often times during the course of a statistical investigation, it is necessary to manipulate your data. ViSta has three data manipulation tools:

1) ViVa, ViSta's Variable Language, creates and modifies variables using statements that are like algebraic equations. ViVa is easy to learn and use, but is limited in scope and can only be used in the listener. It is a special purpose language intended for manipulating ViSta variables.

2) VAR creates and modifies variables using expressions in the Lisp language. VAR is faster and much more general than ViVaand can be entered from the listener or from files. Although VAR is based on Lisp, which is much harder to learn than ViVa, you only need to learn a small subset of Lisp to manipulate variables. But, since Lisp is a general purpose computing language and ViVa is not, VAR can be much more powerful than ViVa.

3) DATASET creates data from a collection of variables. Note that ViVa and VAR create variables which can be further manipulated by ViVa and VAR. However, they cannot be visualized, analyzed or permanently saved until they are placed into a dataset with DATASET.

DOLLAR FUNCTIONS

ViSta has several functions that tell you what variables are available for use by ViVa, VAR and DATASET. These functions all start with the $ character, and so are called "dollar functions".

To find out what variables are available for creating new variables and forming new data objects, type

NAME	INFORMATION
$vars	list of variables in the current data
$all-vars	list of all variables
$data-vars	list of all data object variables
$free-vars	list of all free variables
$NAME-vars	list of all variables in data object NAME

ViVa Example:

ViVa uses algebraic equations to manipulate and create variables. The equations must be enclosed in brackets. For example, if you have measurements of temperature on the Farenheit scale used in the United States and you wish to convert them to the Celsius scale used by the rest of the world, you would type:

```
[celsius = (5 / 9) * (farenheit - 32)]
```

The variable on the left of the equal sign is created and assigned the value of the expression on the right.

VAR Example:

You can also use VAR to manipulate existing variables and calculate new ones. VAR statements have the form:

```
(VAR variable formula)
```

This statement creates a new variable named "variable" whose value is the result of the calculations performed by "formula". Here's how you do the temperature scale conversion example with VAR:

```
(var celsius (* (/ 5 9) (- farenheit 32)))
```

Notice that functions (such as * / and -) always preceed their arguments.

DATASET Example:

DATASET takes a collection of variables and makes a new dataobject from them. The dataobject persists throughout the session with ViSta and can be permanently saved to a file. The variables in the dataset can be visualized and analyzed with ViSta.

For example, to put the two temperature scales in a datafile you type:

```
(dataset both-scales celsius farenheit)
```

The general form of the DATASET statement is

```
(DATASET NAME VAR1 VAR2 ...
         :TYPES '("numeric" "category" ...)
         :LABELS '("A" "B" "C" "D" ...)
         )
```

The :TYPES and :LABELS keywords are used to specify variable types and observation names. These arguments are optional, although you must use :TYPES when you have categorical variables.

ViVa - ViSta's Variable Language (3 of 6)

ViVa - VISTA'S VARIABLE LANGUAGE

ViVa is an algebra-like language which calculates values for new variables and makes the variables available for further manipulation.

ViVa consists of algebra-like expressions enclosed in a pair of brackets. If the expression involves assignment to a variable, the variable is assigned the value of the expression, and the variable is saved for the duration of the session with ViSta.

ViVa can be used in 3 ways. These ways are outlined briefly here. Examples of ViVa expressions are given below.

1) At the Listener, type a ViVa expression enclosed in brackets. For example

```
[y = 2 + 3*x]
```

The expression on the right side of the = sign is evaluated, the resulting value is assinged to y, and the value is returned. Thus:

```
> [x=4.5]
4.5
> [y = 2 + 3*x]
15.5
```

2) At the Listener, type a left bracket and a return. Lisp's > prompt is replaced by ViVa's ? prompt. Then you can type a series of ViVa expressions, each followed by a return. Each expression is evaluated when you type a return. Finally, type a right bracket to return to regular Lisp. The value of the last expression is returned. For example:

```
> [
? s = list(2,4,6)
(2 4 6)
? u = 3
3
? v = 2
2
? r = u + v*s
(7 11 15)
? ]
(7 11 15)
>
```

3) Enter a bracket enclosed ViVa statement in the middle of lisp. It is evaluated and returns its value to Lisp. Thus:

```
> (list 1 2 [sqrt(9)] 4)
(1 2 3.0 4)
```

LIMITATIONS:

1) ViVa can only be entered from the Listener, not from files. Thus, ViVa scripts or applets are not possible.

2) When using ViVa, variable names cannot contain embedded characters interpreted as mathematical operators (i.e., - + = / *, etc). These signs are always interpreted as being a mathematical operator. In particular, the Lisp convention of variable or function names containing

dashes, such as principal-components or normal-rand, is not allowed in ViVa. Underscores may be substituted for dashes (see next point).
3) Underscores may only be used as a substitute for dashes (see previous point).

UNDERSCORES:
Since dashes are very commonly used in the names Lisp functions, you may substitute underscores for dashes. Thus, you can enter principal_components or normal_rand. ViVa will translate appropriately.

VERBOSE:
Type (viva-verbose t) or (viva-verbose nil) at the Lisp > prompt to control the ammount of output generated by ViVa.

VIVA EXPRESSIONS:
ViVa expressions conform to a subset of the syntax for the C programming language. Specifically, ViVa supports all syntax defined in section 18.1 of the original Kernighan and Ritchie book, plus all C numerical syntax including floats and radix syntax (i.e. 0xnnn, 0bnnn, and 0onnn). ViVa supports multiple array subscripts.

ABOUT VIVA AND PARCIL
ViVa uses PARCIL, copyright (c) 1992 by Erann Gat. Parcil is used under the terms of the GNU General Public License as published by the Free Software Foundation. PARCIL parses expressions in the C programming language into Lisp. ViVa then takes the parsed statements and creates an interactive environment in which they are evaluated. Thanks to Sandy Weisberg for providing Parcil.

ViVa EXAMPLES

Examples of each way of using ViVa are given below.

1) At the Listener, type a ViVa expression enclosed in brackets. It is evaluated and the value returned. If the expression involves assignment to a variable, the variable is assigned the value, and the variable is saved for the duration of the session. The variable can be entered into a data object, using the DATASET function, and then saved permanently.
Here is an example:

```
> [A = 2 + (3*4)]
14
```

Here, the user has typed the ViVa expression A=2+(3*4) enclosed in brackets. ViSta responds with 14, the appropriate value, using the standard rules of operator precedence. A new Lisp variable A has been created:

```
> a
14
>
```

You can retrieve the list of all saved variables created by ViVa by typing

```
> $viva-vars
```

These variables, and those created by the VAR function, are free variables (not in a data object). You can get a list of all free variables by typing:

```
> $free-vars
```

You can also retrieve the list of all variables in all data objects by typing:

```
> $data-vars
```

Note that any Lisp function may be used in ViVa expressions, though in standard algrebraic syntax. For example:

```
    > [a=3*iseq(4)]
    (0 3 6 9)
```
where iseq(4) is the ViVa equivalent of Lisp's (iseq 4), which generates the sequence of numbers (0 1 2 3).

These functions can use any $data-vars as arguments:
```
    > [b=a^2]
    > (0 9 36 81)
```
A Lisp expression involving a function with multiple arguments is written as a ViVa function involving arguments that are comma-delimited. Thus
```
    > [L=list(1,2,3)]
```
corresponds to the Lisp expression (setf L (list 1 2 3)).
Note that a ViVa expression may be a compound expression:
```
    > { ( x=1,y=2,print(x+y),sin(pi/2) ) }
    3
    1.0
    > x
    1
    > y
    2
    >
```

2) At the Listener, type a left bracket and a return. Lisp's > prompt is replaced by ViVa's ? prompt. Then you can type a series of ViVa expressions, each being evaluated when you type a return. Finally, type a right bracket to return to regular Lisp. Notice that functions with multiple arguments have their arguments separated by commas. Also, notice that vector and matrix algebra are supported. Here is an example interaction:

```
    > [
    ? 2+3
    5
    ? a
    (0 3 6 9)     ;this value for A is left from above
    ? a=2+3
    5
    ? a           ;a new value for A has been defined
    5
    ? b=sqrt(a)
    2.23606797749979
    ? a=iseq(4)   ;yet another value for A
    (0 1 2 3)
    ? b=sqrt(a)   ;vector arithmetic is supported
    (0.0 1.0 1.4142135623730951 1.7320508075688772)
    ? ]           ;the value of the last expression is returned
    (0.0 1.0 1.4142135623730951 1.7320508075688772)
    >
```

If you do not wish to see so much output, type:
```
    [viva_verbose=not(t)]
```
For example:
```
    > [viva_verbose=not(t)
    ? a=normal_rand(10)
    ? b=a^2
    ? c=iseq(10)
    ? d=c*b
    ? ]
    (0.0 0.12086435903614712 0.12465600120144842 0.05113395472276512
0.8572483580685174 2.8192853102290143 8.009152301820079
```

```
33.31659449692597 0.9436767853187026 1.1345846087676839)
    > d
(0.0 0.12086435903614712 0.12465600120144842 0.05113395472276512
0.8572483580685174 2.8192853102290143 8.009152301820079
33.31659449692597 0.9436767853187026 1.1345846087676839)
    >
   Lisp's matrix language may be used with ViVa. Here is an example:
   ?a=matrix ( list(2,2), list(1, 2 ,3 ,4))
   #2A((1 2) (3 4))
   ?b=matrix ( list(2,3), list(1,2,3,4,5,6))
   #2A((1 2 3) (4 5 6))
   ?c=matmult(a,b)
   #2A((9.0 12.0 15.0) (19.0 26.0 33.0))
```

3) Enter a bracket enclosed ViVa statement in the middle of lisp. It is evaluated and returns its value to Lisp:

```
   >(list 1 3 [sqrt(25)] 7)
   (1 3 5.0 7)
```

TWO EXAMPLES

The first example involves calculating points in an Introductory Statistics course. The data are in P3099pts.lsp datafile. You can open these data with the OPEN DATA menu item in the FILE menu. In this example, ViVa is used to create variables, then the DATASET function is used to create a data object from them.

```
[
? homework_sum=homework1+homework2=homework3
(12.0 14.24 13.399999999999999 13.11 15.36 12.96 12.22 13.82 14.31 13.09
13.98 13.67 7.45 15.33 12.1 12.92 5.57)
? total=homework_sum+midterm1
(170.0 173.24 168.4 189.11 197.36 119.96 159.22 135.82 194.31 189.09
197.98 189.67 159.45 194.33 153.1 176.92 173.57)
? ]
(170.0 173.24 168.4 189.11 197.36 119.96 159.22 135.82 194.31 189.09
197.98 189.67 159.45 194.33 153.1 176.92 173.57)
> (dataset summary homework_sum total)
#<Object: a9e82c, prototype = MV-DATA-OBJECT-PROTO>
>
```

The second example involves calculating grades from points earned during the course.

```
(load (strcat *data-dir-name* "p3099pts.lsp"))

[homeworksum = homework1 + homework2 + homework3]
[midpts = homeworksum + midterm1]

(let ((grades (mapcar #'(lambda (pts)
          (cond ((> pts 190) "A")
                ((< 170 pts 190) "B")
                ((< 150 pts 170) "C")
                ((< 140 pts 150) "C-")
                ((<   0 pts 150) "D")))
                          midpts))))

(dataset homework1 homework2 homework3 homeworksum
         midterm1 midpts grades)
```

The VARIABLE Maker (4 of 6)

VAR - ViSta's Variable Maker

VAR creates or reports the values of a ViSta variable object. When creating a variable object, the name of the new variable object is added to the $free-vars and $viva-vars lists.

SYNTAX: (VAR VARIABLE &OPTIONAL FORM &KEY (TYPE NUMERIC))

Usage: var can be used in three ways:
1: (VAR VARIABLEe) to report the value of variable
   Example: (var midterm-correct)
2: (VAR VARIABLE FORM) to bind variable to the result of form
   Example: (var final-points (* 2 (list 78 45 67 93 89)))
3: (VAR VARIABLE FORM :TYPE TYPESYMBOL) like 2, but non-numeric variable
   Example: (var grades '(b d c a a-) :type category)

VARIABLE is not evaluated and must be a symbol.

When FORM is specified, it is evaluated, returned, and bound to VARIABLE, with VARIABLE being appended to $DATA-VARS, the list of all ViSta variables and $DESK-VARS, the list of all ViSta variables that are not in a dataobject. Also, when FORM is specified a variable VARIABLE-O is created and bound to the object-id of VARIABLE.

When FORM is not specified, returns the value of VARIABLE if it is on the $DATA-VARS or $DESK-VARS list. Produces an error if not.

VARIABLE will be NUMERIC unless TYPE is the symbol ORDINAL, CATEGORY, FREQ or LABEL. TYPE cannot be specified unless FORM is specified. If VARIABLE is already bound and the global variable *ASK-ON-REDEFINE* is not nil then you are asked if you want to redefine the variable.

DATA - The DATA Maker (5 of 6)

DATA OBJECTS AND THE DATASET MACRO

DATA OBJECTS are created by the DATASET macro from a collection of variables. DATASET can also report the names of data objects and can be used for more advanced data manipulation as well. We discuss the macro in this help file.

CREATING DATA OBJECTS

DATASET is used to create a data object from a collection of variables. This is done by typing:

```
(DATASET NAME VAR1 VAR2 ...)
```

where NAME is the name of the new data object being created, and VAR1 VAR2, etc, are the names of numeric variables. When the variables are not all numeric, you must include the :TYPES keyword, followed by the types:

```
(DATASET NAME VAR1 VAR2 :TYPES (NUMERIC CATEGORY))
```

The variables must all have the same number of observations. You may use long or short names (see the ABOUT VARIABLES AND DATA help file). While short names are easier to type, they introduce the possibility of duplicate variable names. If there are duplicates, the most recently created variable is usually used, though ambiguity is possible. Long names, while more cumbersome, prevent the possibility of duplicate names. The variables may be "free" variables (i.e., those which were created by ViVa or VAR) or "data" variables (those already in data objects). Variables which are in more than one data object are distinct variables: Changes made to one will not cause changes in the other.

To find out what variables are available to form a new data object, type

```
NAME         INFORMATION
$vars        list of variables in the current data
$all-vars    list of all variables
$data-vars   list of all data object variables
$free-vars   list of all free variables
$NAME-vars   list of all variables in data object NAME
```

You may label your observations by using the :LABELS keyword, which must be followed by a list of strings specifying observation names ("Obs1", "Obs2", etc., by default).

```
(DATASET NAME VAR1 VAR2
         :TYPES (NUMERIC CATEGORY)
         :LABELS ("A" "B" "C"))
```

If the data are frequency data, you must use the :FREQ keyword followed by T (for true) to specify that the values of the numeric variables are frequencies. For example

```
(dataset freqdata frequency center treatment
         :types (numeric category category)
         :freqs t)
```

You may use the :ABOUT keyword, followed by an optional string of information about the data.

Given the arguments discussed above you can specify:
1) MULTIVARIATE data are data which are not one of the other data types given below. These data include univariate (one variable) and bivariate (two variables) data.
2) CATEGORY data have one or more CATEGORY variables and no NUMERIC or ORDINAL variables. The N category variables define an n-way classification.
3) CLASSIFICATION data have one NUMERIC variable and one or more CATEGORY variables. The N category variables define an n-way classification. The numeric variable specifies an observation for a given classification.
4) FREQUENCY CLASSIFICATION data are classification data whose numeric variable specifies frequencies as indicated by using FREQ. The N category variables define an n-way classification, with the numeric variable specifying the co-occurance frequency of a specific combination of categories.

ADVANCED USES: FREQUENCY TABLE DATA AND MATRIX DATA

Frequency table data are data whose observations and variables are used to form the rows and columns of a two-way table. That is, the data are a two-way cross tabulation of the co-occurance frequency formed from the observations and variables. The data elements must be frequencies. The variables must be NUMERIC and :FREQ T must be specified. In addition, :ROW-LABEL and :COLUMN-LABEL must be used. Each must be followed by a string. The string is used to label the rows or columns of the table.

MATRIX data are data whose observations and variables refer to the same things. These things are used to form a square, usually symmetric matrix with the same number of rows and columns, the rows and columns identifying the same things. The values might be correlations, covariances, distances, etc. Optionally, there can be more than one matrix in a given data object. All matrices must have rows and columns identifying the same things. The keyword :MATRICES, used only for matrix data, must be followed by a list of strings specifying matrix names (and, indirectly, the number of matrices). :SHAPES, optional for matrix data only, must be a list of strings "symmetric" or "asymmetric" (case ignored), specifying the shape of each matrix (all are symmetric by default).

THE LONG-FORM OF THE DATASET MACRO

The complete "long-form" of the DATASET function creates a new data object from information contained within the DATASET statement. The minimum required syntax is:

```
(DATASET NAME
         :VARIABLES (VARLIST)
         :DATA (DATALIST) )
```

For example:

```
(dataset example
         :variables (abc def)
         :data (1 2 3 4 5 6))
```

The required arguments are discussed next, with optional long-form arguments following:

REQUIRED ARGUMENTS:
NAME &KEY :DATA :VARIABLES

NAME must be a string or a symbol. This is the name of the newly defined data object.

:VARIABLES must be followed by a list of strings or symbols defining variable names (and, indirectly, the number of variables).

:DATA must be followed by a list of numbers, strings or symbols (symbols are converted to uppercase strings). The number of data elements must conform to the information in the other arguments.

GENERAL OPTIONAL ARGUMENTS:
&KEY :TYPES :LABELS :FREQ :ABOUT

:TYPES must be followed by a list of strings "numeric", "ordinal" or "category" (case ignored), or symbols (same as strings, but no quotes) specifying whether the variables are numeric, ordinal or categorical (all numeric by default).

:LABELS must be followed by a list of strings specifying observation names ("Obs1", "Obs2", etc., by default).

:FREQ must be followed by T to specify that the values of the numeric variables are frequencies.

:ABOUT is followed by an optional string of information about the data.

ADDITIONAL USES FOR THE DATASET MACRO

The dataset macro may also be used to find out information about data objects. In particular, to see a list of all data objects, type:

```
(DATASET)
```

and to see the object identification of data object NAME, type:

```
(DATASET NAME)
```

Finally, you can create a new data object from a program contained within the DATASET macro by typeing:

```
(DATASET NAME FORM)
```

where NAME is the name of the new data object, and FORM is a Lisp form, as described by Tierney (1990) or Steele (1990).

ViSta - The Visual Statistics System
Wednesday, December 13, 2000 - 8:21 AM

The DATACODE-editor (6 of 6)

If you wish, you may also enter your data by using a text editor to create a text file in a specific format. The format is described in the ViSta User's Guide. You can get a good idea of the format by looking at any file in one of the data folders. Briefly, the minimum format for multivariate data is:

```
(data "name"
:variables '("Variable A" "Variable B" "Variable C")
:data '(
1 2 3
4 5 6
7 8 9
10 11 12
))
```

Additional features are described in the User's Guide.

The DATA function creates a new data object or reports the names of all data objects or the object identifcation of a specific object. It can be used in three different ways:
1) To see a list of all data objects, type:
   (DATA)
2) To see the object identification of data object NAME, type:
   (DATA NAME)
3) To create a new data object from information contained within the DATA statement, the minimum syntax requires you to type:
   (DATA 'NAME :VARIABLES <VARLIST> :DATA <DATALIST>)

GENERAL ARGUMENTS:
&OPTIONAL NAME &KEY DATA VARIABLES TYPES LABELS FREQ ABOUT
Defines ViSta data object NAME. NAME, DATA and VARIABLES are required arguments.

NAME must be a string or a symbol. If a symbol it must be preceeded by a single quote.

DATA must be followed by a list of numbers, strings or symbols (or mix of such). Symbols are converted to uppercase strings. The number of data elements must conform to the information in other arguments.

VARIABLES must be followed by a list of strings or a single quoted list of symbols defining variable names (and, indirectly, the number of variables).

TYPES, optional, must be a list of the strings \"numeric\", \"ordinal \" or \"category\" (case ignored), or a list of the corresponding symbols. These strings or symbols specify whether the variables are numeric, ordinal or categorical (all numeric by default). Note that the oridinal datatype is seldomly used.

LABELS, optional, must be a list of strings or symbols specifying observation names (\"Obs1\", \"Obs2\", etc., by default). Symbols are converted to uppercase strings.

FREQ specifies that the values of the numeric variables are

frequencies.

ABOUT is an optional string of information about the data.

Given these arguments above you can specify the following types of data:
1) MULTIVARIATE data are data which are not one of the other data types given below. These data include univariate (one numeric or ordinal variable) and bivariate (two numeric or ordinal variables) data.
2) CATEGORY data have one or more CATEGORY variables and no NUMERIC or ORDINAL variables. The N category variables define an n-way classification.
3) CLASSIFICATION data have one NUMERIC variable and one or more CATEGORY variables. The N category variables define an n-way classification. The numeric variable specifies an observation for a given classification.
4) FREQUENCY CLASSIFICATION data are classification data whose numeric variable specifies frequencies as indicated by using FREQ. The N category variables define an n-way classification, with the numeric variable specifying the co-occurance frequency of a specific combination of categories.

ARGUMENTS FOR FREQUENCY TABLE DATA:
ARGUMENTS: &KEY ROW-LABEL, COLUMN-LABEL, and FREQ
For FREQUENCY TABLE data, the ROW-LABEL and COLUMN-LABEL arguments must be used: These data have NUMERIC variables whose values specify frequencies as indicated by using FREQ. The data are a two-way cross tabulation of the co-occurance frequency of the row and column entities. The ROW-LABEL and COLUMN-LABEL identify the data as two-way array (table) data, with the numeric variables in the data represent columns of a two-way table rather than variables of a multivariate dataset. ROW-LABEL and COLUMN-LABEL each have a string value which labels the rows or columns (ways) of the array. Observation labels are used to label the row-levels and variable names the column-levels.

ARGUMENTS FOR MATRIX DATA
ARGUMENTS: &KEY MATRICES SHAPES
These arguments are used to identify matrix data. MATRICES, required for matrix data only, must be a list of strings specifying matrix names (and, indirectly, the number of matrices). SHAPES, optional for matrix data only, must be a list of strings \"symmetric\" or \"asymmetric\" (case ignored), specifying the shape of each matrix (all are symmetric by default). Matrix arguments cannot be used with array data.

EFFICIENCY ARGUMENTS: &KEY MISSING-VALUES, STRINGS
1) If you know the data do or do not contain missing values, specifying MISSING-VALUES as T or NIL eliminates the time required to check for missing values.
2) Specify STRINGS T if you know that all category values are represented by strings (values inside double-quote marks) then the need to check for non-string category values is eliminated, greatly increasing efficiency, especially for large data.

The preceeding arguments are all of those concerning the definition of data within ViSta. There are additional arguments which concern the creation of data by programming. These are described below.

ARGUMENTS FOR PROGRAMMING
ARGUMENTS: &KEY PROGRAM, USE

If you wish to write a data program, use these arguments:
1) PROGRAM (required) specifies that a program follows which computes N new variables. The program must return a list of N lists corresponding to the N variables in the :VARIABLES keyword. Causes the new variables to be bound.
2) USE (optional) specifies the data object whose variables are input to the program. This must be a data object with bound variables. If not, specify :USE (BIND-VARIABLES DOB)

For example, assume there is a dataobject named pcaexmpl containing variables X and Y. Then you can type: .

```
(data "PCAExmpl2"
      :use pcaexmpl
      :variables '("X" "Y" "A" "B" "C")
      :program
       (let* ((A (+       X  Y))
              (B (+ (*  5 X) Y))
              (C (+ (* -2 X) Y)))
      (list x y a b c)))
```

Which creates a new data object containing variables a b and c as well as the
original x and y.

Consider this example (which assumes the variables hmw1 through hmwk4 already exist in the points dataobject):

```
(data "WeightedPoints"
      :use points
      :program
(let* ((h1 (* hmwk1 5/10))
       (h2 (+ (* hmwk2a 6/10)    (* hmwk2b 4/10)))
       (h3 (+ (* hmwk3abd 7/10) (* hmwk3c 3/10)))
       (h4 (* hmwk4 15/10))
       (hmwksum (+ h1 h2 h3 h4))
       (total (+ hmwksum midterm1 attend vis)))
  (list attend  vis h1 h2 h3 h4 hmwksum total))
      :variables '("attend" "visual" "h1" "h2" "h3" "h4" "htot"
"total"))
```

This program takes the variables hmw1 through hmwk4 in the points dataobject and
combines them together for a total homework score, plus adding other variables
in SCORE to get a point total for a class.

Finally, there are arguments that effect the operation of the system. These should only be used by knowledgeable system developers. These arguments are:

&KEY CREATED CREATOR-OBJECT SUBORDINATE ICONIFY DATASHEET-ARGUMENTS NEW-DATA ALL-TYPES-IN-DATA-ARRAY

15% off

any Bull's Head item in stock

expires May 31, 2001

UNC Student Stores — UNC Student Stores — UNC Student Stores

FREE Medium Fountain Drink

at one of these snack units:

Circus room, Blue Ram, Nook, or Osler

expires May 31, 2001

UNC Student Stores — UNC Student Stores — UNC Student Stores

FREE Cup of Coffee Regular Size

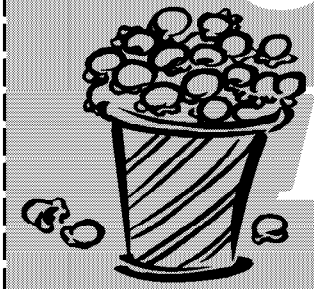
at the **Pit Stop**

(1st floor Student Stores)

expires May 31, 2001

UNC Student Stores — UNC Student Stores — UNC Student Stores

FREE Box of Popcorn



at the **Pit Stop**

(1st floor Student Stores)

expires May 31, 2001

UNC Student Stores — UNC Student Stores — UNC Student Stores